COSC-6590/GSCS-6390

Games: Theory and Applications

Lecture 03 - Zero-Sum Matrix Games

Luis Rodolfo Garcia Carrillo

School of Engineering and Computing Sciences
Texas A&M University - Corpus Christi, USA

September 18, 2018

Table of contents

Zero-Sum Matrix Games

Zero-Sum Matrix Games

Played by two players, each having available a finite set of actions (an **action space**):

- P_1 has available m actions: $\{1, 2, \ldots, m\}$
- P_2 has available n actions: $\{1, 2, \ldots, n\}$

The **outcome** J is quantified by an $m \times n$ matrix $A = [a_{ij}]$.

- entry a_{ij} provides the outcome of the game when

$$\begin{cases} P_1 \text{ selects action } i \in \{1, 2, \ldots, m\} \\ P_2 \text{ selects action } j \in \{1, 2, \ldots, n\} \end{cases}$$

Note. One can imagine that

- P_1 selects a row of A.
- P_2 selects a column of A.

Zero-Sum Matrix Games

Objective (zero sum). P_1 wants to **minimize** the outcome J, and P_2 wants to **maximize** J.

- P_1 is called the minimizer. It selects the **rows**.
- P_2 is called the maximizer. It selects the **columns**.

The **outcomes** is called

- a **cost**, from P_1's perspective
- a **reward**, from P_2's perspective

Example 3.1. The matrix A defines a zero-sum matrix game for which: minimizer has 3 actions, maximizer has 4 actions.

$$A = \underbrace{\left.\begin{bmatrix} 1 & 3 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ -2 & 2 & 0 & 1 \end{bmatrix}\right\}}_{P_2 \text{ choices}} P_1 \text{ choices}$$

Security Levels and Policies

Security Levels and Policies

Secure (**risk averse**) playing

- choices made by a player, guaranteed to produce the best outcome against **any** choice made by the other player (**rational or not**).

For the matrix game in **Example 3.1** the following are secure policies for each player

P_2**: column 3 is a security policy**

- it **guarantees a reward of at least** 0, and
- no other choice can guarantee a larger reward

P_1**: Rows 2 and 3 are security policies**

- they both **guarantee a cost no larger than 2**, and
- no other choice can guarantee a smaller cost.

Security Levels and Policies

Definition 3.1 (Security policy).

Consider a matrix game defined by the matrix A.

The **security level** for P_1 (the minimizer) is defined by

$$\bar{V}(A) := \underbrace{\min_{i\in\{1,2,\ldots,m\}}}_{\substack{\text{minimize cost assuming} \\ \text{worst choice by } P_2}} \underbrace{\max_{j\in\{1,2,\ldots,n\}}}_{\substack{\text{worst choice by } P_2 \\ \text{from } P_1\text{'s perspective}}} a_{ij}$$

MATLAB® Hint 1.

Compute P_1's security level using

```
min(max(A))
```

Security Levels and Policies

The corresponding **security policy** for P_1

- any i^* that achieves the desired security level, i.e.,

$$\underbrace{\max_{j \in \{1,2,\ldots,n\}} a_{i^*j} = \bar{V}(A)}_{i^* \text{ achieves the inimum}} := \min_{i \in \{1,2,\ldots,m\}} \max_{j \in \{1,2,\ldots,n\}} a_{ij}$$

Notation. This equation is often written as

$$i \in \arg\min_i \max_j a_{ij}.$$

The use of "∈" instead of "=" emphasizes that there may be several i^* that achieve the minimum.

Security Levels and Policies

The **security level** for P_2 (the maximizer) is

$$\underline{V}(A) := \underbrace{\max_{j\in\{1,2,\ldots,n\}}}_{\substack{\text{maximize reward assuming}\\ \text{worst choice by } P_1}} \underbrace{\min_{i\in\{1,2,\ldots,m\}}}_{\substack{\text{worst choice by } P_1\\ \text{from } P_2\text{'s perspective}}} a_{ij}$$

The corresponding **security policy** for P_2

- any j^* that achieves the desired security level, i.e.,

$$\underbrace{\min_{i\in\{1,2,\ldots,m\}} a_{ij^*} = \underline{V}(A)}_{j^* \text{ achieves the maximum}} := \max_{j\in\{1,2,\ldots,n\}} \min_{i\in\{1,2,\ldots,m\}} a_{ij}$$

Notation. This equation is often written as

$$j \in \arg\max_j \min_i a_{ij}.$$

Security Levels and Policies

In view of the reasoning above, for the matrix A

$$A = \underbrace{\left.\begin{bmatrix} 1 & 3 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ -2 & 2 & 0 & 1 \end{bmatrix}\right\} P_1 \text{ choices}}_{P_2 \text{ choices}}$$

we have that security levels are

$$\underline{V}(A) = 0 \leq \bar{V}(A) = 2$$

Note: The letter V stands for **value**.

Security Levels and Policies

Security levels/policies satisfy the following three properties:

Proposition 3.1 (Security levels/policies)

For every (finite) matrix A, the following properties hold:

P3.1 Security levels are well defined and unique.

P3.2 Both players have security policies (not necessarily unique).

P3.3 The security levels always satisfy the inequalities

$$\underline{V}(A) := \max_{j \in \{1,2,\ldots,n\}} \min_{i \in \{1,2,\ldots,m\}} a_{ij} \leq \bar{V}(A) := \min_{i \in \{1,2,\ldots,m\}} \max_{j \in \{1,2,\ldots,n\}} a_{ij}$$

The advertising campaign: Simultaneous Play

Properties **P3.1** and **P3.2** are trivial from the definitions.

P3.3 follows from the following reasoning.

Let j^* be a security policy for the maximizer P_2, i.e.,

$$\underline{V}(A) = \min_i a_{ij^*}$$

Since

$$a_{ij^*} \leq \max_j a_{ij}, \qquad \forall i \in \{1, 2, \ldots, m\}$$

we conclude that

$$\underline{V}(A) = \min_i a_{ij^*} \leq \min_i \max_j a_{ij} =: \bar{V}(A)$$

which is precisely what **P3.3** states.

Security Levels/Policies with MATLAB

Computing Security Levels and Policies with MATLAB

MATLAB® Hint 1 (`min` and `max`).

Either of the commands

```
[Vover,i]= min(max(A,[],2))      [Vover,i ]= min(max(A))
```

compute the security level `Vover` and a security policy `i` for P_1. Maximization is along the second dimension `A`: `[],2`

Either of the commands

```
[Vunder,j]= max(min(A,[],1))     [Vunder,j]= max(min(A))
```

compute the security level `Vunder` and a security policy `j` for P_2. Minimization is along the first dimension `A`: `[],1`

When more than one security policies exist, the one with the lowest index is returned.

Security vs. Regret

Security vs. Regret: Alternate Play

Suppose that the minimizer P_1 plays first ($P_1 - P_2$ game).

For the matrix game in **Example 3.1**

$$A = \underbrace{\begin{bmatrix} 1 & 3 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ -2 & 2 & 0 & 1 \end{bmatrix}}_{P_2 \text{ choices}} \left.\vphantom{\begin{bmatrix} 1 \\ 0 \\ -2 \end{bmatrix}}\right\} P_1 \text{ choices}$$

the optimal policy for P_2 (maximizer) is

$$\pi_2 \equiv P_2 \text{ selects } \begin{cases} \text{column 2 (or 3)} & \text{if } P_1 \text{ selected row 1, leading to a reward of 3} \\ \text{column 3} & \text{if } P_1 \text{ selected row 2, leading to a reward of 2} \\ \text{column 2} & \text{if } P_1 \text{ selected row 3, leading to a reward of 2} \end{cases}$$

in view of this, the optimal policy for P_1 (minimizer) is

$$\pi_1 \equiv P_1 \text{ selects row 2 (or 3), leading to a cost of 2}$$

Security vs. Regret: Alternate Play

If both players are rational, the outcome is the security level for the player that plays first (P_1 in this case)

$$\bar{V}(A) = 2$$

and no player will regret their choice after the games end.

If the maximizer P_2 plays first ($P_2 - P_1$ game), the outcome is the security level for the player that plays first (P_2 in this case):

$$\underline{V}(A) = 0$$

and again no player will regret their choice after the games end.

Conclusion: For any **matrix game with alternate play** there is no reason for rational players to ever regret their decision to **play a security policy**.

Security vs. Regret

Security vs. Regret: Simultaneous Plays

Suppose P_1 and P_2 must decide on their actions simultaneously (without knowing the others choice)

$$A = \underbrace{\left.\begin{bmatrix} 1 & 3 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ -2 & 2 & 0 & 1 \end{bmatrix}\right\} P_1 \text{ choices}}_{P_2 \text{ choices}}$$

If both players use their respective security policies then

$$\begin{cases} P_1 \text{ selects row 3,} & \text{guarantees cost } \leq 2 \\ P_2 \text{ selects column 3,} & \text{guarantees reward } \geq 0 \end{cases}$$

leading to cost/reward $= 0 \in [\underline{V}(A), \bar{V}(A)]$

Security vs. Regret: Simultaneous Plays

$$A = \underbrace{\begin{bmatrix} 1 & 3 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ -2 & 2 & 0 & 1 \end{bmatrix}}_{P_2 \text{ choices}} \left.\vphantom{\begin{bmatrix} 1 \\ 0 \\ -2 \end{bmatrix}}\right\} P_1 \text{ choices}$$

After the game is over…

- P_1 is **happy**: row 3 was the best response to column 3
- P_2 has **regrets**: "if I knew P_1 was going to play row 3, I would have played column 2, leading to reward $= 2 \geq 0$"

Perhaps they should have played

$$\begin{cases} P_1 \text{ selects row 3,} \\ P_2 \text{ selects column 2,} \end{cases} \qquad \text{leading to cost/reward } = 2$$

Security vs. Regret: Simultaneous Plays

Now the **minimizer regrets** its choice!

No further **a-posteriori** revision of the decisions would lead to a no-regret outcome.

Important observation:

(As opposed to what happens in alternate play)

Security policies may lead to regret in matrix games with simultaneous play.

Saddle-Point Equilibrium

Saddle-Point Equilibrium

Example 3.2. A defines a zero-sum matrix game in which both minimizer and maximizer have 2 actions:

$$A = \underbrace{\left.\begin{bmatrix} 3 & 1 \\ -1 & 1 \end{bmatrix}\right\}}_{P_2 \text{ choices}} P_1 \text{ choices}$$

For this game

- P_1's security level is $\bar{V}(A) = 1$
  - the corresponding security policy is row 2
- P_2's security level is $\underline{V}(A) = 1$
  - the corresponding security policy is column 2.

Saddle-Point Equilibrium

If both players use their **security policies**

$$\begin{cases} P_1 \text{ selects row 2,} & \text{guarantees cost } \le 1 \\ P_2 \text{ selects column 2,} & \text{guarantees reward } \ge 1 \end{cases}$$

leading to cost/reward $= 1 = \underline{V}(A) = \bar{V}(A)$

No player regrets their choice

- their policy was optimal against what the other did.

Same result would have been obtained in an **alternate play game** regardless of who plays first.

Lack of regret: one is not likely to change ones policy in subsequent games, leading to a **stable** behavior.

Saddle-Point Equilibrium

Definition 3.2 (Pure saddle-point equilibrium)

Consider a matrix game defined by the matrix A.

A pair of policies (i^*, j^*) is called a (**pure**) **saddle-point equilibrium** if

$$a_{i^*j^*} \leq a_{ij^*} \qquad \forall i$$

$$a_{i^*j^*} \geq a_{i^*j} \qquad \forall j$$

and $a_{i^*j^*}$ is called the (**pure**) **saddle-point value**.

These equations are often re-written as

$$a_{i^*j} \leq a_{i^*j^*} \leq a_{ij^*} \qquad \forall i, j$$

and also as

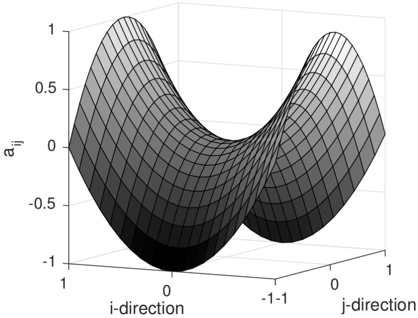
$$a_{i^*j^*} = \min_i a_{ij^*} \qquad a_{i^*j^*} = \max_j a_{i^*j}$$

Saddle-Point Equilibrium

$a_{i^*j^*}$ increases along the "i-direction"

$a_{i^*j^*}$ decreases along the "j-direction"

This corresponds to a surface that looks like a horse's saddle.



Saddle-Point Equilibrium

Note 4 (Saddle-point equilibrium).

Equation $a_{i^*j^*} \leq a_{ij^*} \qquad \forall i$

should be interpreted as

- i^* is the best option for P_1 assuming that P_2 plays j^*,

Equation $a_{i^*j^*} \geq a_{i^*j} \qquad \forall j$

should be interpreted as

- j^* is the best option for P_2 assuming that P_1 plays i^*,

These statements could be restated as
"no player will regret her choice, if they both use these policies"
or
"no player will benefit from an unilateral deviation from the equilibrium".

Saddle-Point Equilibrium vs. Security Levels

Saddle-Point Equilibrium vs. Security Levels

The existence of a pure saddle-point equilibrium is related to the security levels for the two players:

Theorem 3.1 (Saddle-point equilibrium vs. security levels).

A matrix game defined by A has a saddle-point equilibrium **if and only if**

$$\underline{V}(A) := \max_{j\in\{1,2,\ldots,n\}} \min_{i\in\{1,2,\ldots,m\}} a_{ij} = \min_{i\in\{1,2,\ldots,m\}} \max_{j\in\{1,2,\ldots,n\}} a_{ij} =: \bar{V}(A)$$

In particular,

1. if (i^*, j^*) is a saddle-point equilibrium then i^* and j^* are security policies for P_1 and P_2, respectively and the equation is equal to the saddle-point value;
2. if the equation holds and i^* and j^* are security policies for P_1 and P_2, respectively then (i^*, j^*) is a saddle-point equilibrium and its value is equal to the equation. □

Saddle-Point Equilibrium vs. Security Levels

Justification of **Theorem 3.1**

If there exists at least one saddle-point equilibrium then the equation must hold.

Assume (i^*, j^*) is a saddle-point equilibrium, then

$$a_{i^*j^*} = \min_i a_{ij^*} \underbrace{\leq}_{\text{since } j^* \text{ is one particular } j} \max_j \min_i a_{ij} =: \underline{V}(A)$$

Similarly

$$a_{i^*j^*} = \max_j a_{i^*j} \underbrace{\geq}_{\text{since } i^* \text{ is one particular } i} \min_j \max_i a_{ij} =: \bar{V}(A)$$

Therefore

$$\bar{V}(A) \leq a_{i^*j^*} \leq \underline{V}(A)$$

Saddle-Point Equilibrium vs. Security Levels

But we saw that for **every** matrix A: $\underline{V}(A) \leq \bar{V}(A)$

All these inequalities are only possible if: $\bar{V}(A) = a_{i^*j^*} = \underline{V}(A)$

Confirming the equation must hold when a saddle point exists.

In addition, since

$$a_{i^*j^*} = \min_i a_{ij^*} \underbrace{\leq}_{\text{since } j^* \text{ is one particular } j} \max_j \min_i a_{ij} =: \underline{V}(A)$$

holds with equality, we conclude that j^* must be a security policy for P_2 and since

$$a_{i^*j^*} = \max_j a_{i^*j} \underbrace{\geq}_{\text{since } i^* \text{ is one particular } i} \min_j \max_i a_{ij} =: \bar{V}(A)$$

holds with equality, i^*must be a security policy for P_1.

Saddle-Point Equilibrium vs. Security Levels

Now show that when the equation holds, a saddle-point equilibrium always exists.

The saddle-point equilibrium can be constructed by taking a security policy i^* for P_1 and j^* for P_2.

Since i^* is a security policy for P_1, we have that

$$\max_j a_{i^*j} = \bar{V}(A) \quad \left(:= \min_j \max_i a_{ij}\right)$$

Since j^* is a security policy for P_2, we also have that

$$\min_i a_{ij^*} = \underline{V}(A) \quad \left(:= \max_j \min_i a_{ij}\right)$$

Saddle-Point Equilibrium vs. Security Levels

Because of what it means to be a min/max, we have that

$$\underline{V}(A) := \min_i a_{ij^*} \leq a_{i^*j^*} \leq \max_j a_{i^*j} =: \bar{V}(A)$$

When the equation holds, these two quantities must be equal.

In particular

$$a_{i^*j^*} = \max_j a_{i^*j} \Rightarrow a_{i^*j^*} \geq a_{i^*j}, \quad \forall j$$

$$a_{i^*j^*} = \min_i a_{ij^*} \Rightarrow a_{i^*j^*} \leq a_{ij^*}, \quad \forall i$$

Conclusion: (i^*, j^*) is indeed a saddle-point equilibrium.

Order Interchangeability

Order Interchangeability

Suppose a matrix game defined A has two distinct saddle-point equilibria:

$$(i_1^*, j_1^*) \qquad \text{and} \qquad (i_2^*, j_2^*)$$

In view of **Theorem 3.1**, both have exactly the same value $V(A) = \underline{V}(A) = \bar{V}(A)$, and

- i_1^* and i_2^* are **security policies** for P_1
- j_1^* and j_2^* are **security policies** for P_2

From **Theorem 3.1** we conclude that the mixed pairs

$$(i_1^*, j_2^*) \qquad \text{and} \qquad (i_2^*, j_1^*)$$

- are also saddle-point equilibria
- have the same values as the original saddle points.

Order Interchangeability

Proposition 3.2 (Order interchangeability).

If (i_1^*, j_1^*) and (i_2^*, j_2^*) are saddle-point equilibria for matrix game A, then (i_1^*, j_2^*) and (i_2^*, j_1^*) are also saddle-point equilibria for A, and all equilibria have exactly the same value.

When one of the players finds a saddle-point equilibria (i_1^*, j_1^*) it is irrelevant to them whether or not the other player is playing at the same saddle-point equilibria, because

1. This player will always get the same cost regardless of what saddle-point equilibrium was found by the other player.
2. Even if the other player found a different saddle-point equilibrium (i_2^*, j_2^*), there will be no regrets since the game will be played at a (third) point that is still a saddle-point.

Computational Complexity

Computational Complexity

Suppose we want to minimize a function $f(i)$ defined over a discrete set $\{1, 2, \dots, m\}$

i	1	2	3	$\cdots$	m
$f(i)$	$f(1)$	$f(2)$	$f(3)$	$\cdots$	$f(m)$

Number of operations needed to find the minimum of f: $n - 1$

- one starts by comparing $f(1)$ with $f(2)$,
- then comparing the smallest of these with $f(3)$,
- and so on…

If one suspects that a particular i^* may be a minimum of $f(i)$, one needs to perform exactly $n - 1$ comparisons to verify it.

Computational Complexity

Suppose we want to find security policies from an $m \times n$ matrix

$$A = \underbrace{\begin{bmatrix} & \vdots & \\ \cdots & a_{ij} & \cdots \\ & \vdots & \end{bmatrix}}_{n \text{ choices for} P_2 \text{ (mazimizer)}} \left.\vphantom{\begin{bmatrix} \vdots \\ a \\ \vdots \end{bmatrix}}\right\} m \text{ choices for } P_1 \text{ (minimizer)}$$

To find a **security policy** for P_1 one needs to perform:

1. m maximizations of a function with n values: one for each possible choice of P_1 (row)
2. one minimization of the function of m values that results from the maximizations.

Computational Complexity

Number of operations to find a security policy for P_1 is

$$m(n-1)+m-1=mn-1,$$

The same number is needed to find a security policy for P_2.

Suppose one is given a candidate saddle-point equilibrium (i^*, j^*) for the game. To verify that this pair of policies is a saddle-point equilibrium, verify the saddle-point conditions

$$\begin{aligned} a_{i^*j^*} &\leq a_{ij^*} \qquad \forall i \\ a_{i^*j^*} &\geq a_{i^*j} \qquad \forall j \end{aligned}$$

which only requires $m-1+n-1=m+n-2$ comparisons.

If this test succeeds, we automatically obtain the two security policies (with far fewer comparisons).

Computational Complexity

Example 3.3 Security policy for partially known matrix game

Consider a zero-sum matrix game for which:
- minimizer has 4 actions, maximizer has 6 actions.
- "?": entries of the matrix that are not known.

$$A = \underbrace{\begin{bmatrix} ? & ? & ? & 2 & ? & ? \\ ? & ? & ? & 3 & ? & ? \\ -1 & -7 & -6 & 1 & -2 & -1 \\ ? & ? & ? & 1 & ? & ? \end{bmatrix}}_{P_2 \text{ choices}} \left.\vphantom{\begin{bmatrix} ? \\ ? \\ ? \\ ? \end{bmatrix}}\right\} P_1 \text{ choices}$$

Although we only know 9 out of the 24 entries, we know that

- the value of the game is equal to $V(A) = 1$
- row 3 is a security policy for P_1
- column 4 is a security policy for P_2.

Computational Complexity

Attention!

Having a **good guess** for a saddle-point equilibrium, perhaps **coming from some heuristics or insight** into the game, **can significantly reduce the computation**.

Even if the **guess** comes from heuristics that cannot be theoretically justified, one can answer precisely the question of whether or not the pair of policies is a saddle-point equilibrium and thus whether or not we have security policies, with a relatively small amount of computation.

Practice Exercises

Practice Exercises: Pure security levels/policies

Exercise 3.1. The matrix A defines a zero-sum matrix game

$$A = \underbrace{\left.\begin{bmatrix} -2 & 1 & -1 & 1 \\ 2 & 3 & -1 & 2 \\ 1 & 2 & 3 & 4 \\ -1 & 1 & 0 & 1 \end{bmatrix}\right\} P_1 \text{ choices}}_{P_2 \text{ choices}}$$

Compute the security levels, all security policies for both players, and all pure saddle-point equilibria (if they exist).

Solution. For this game

$\underline{V}(A) = 1$ columns $\{2,4\}$ are security policies for P_2

$\bar{V}(A) = 1$ rows $\{1,4\}$ are security policies for P_1

Game has 4 pure saddle-point equilibria (1,2), (4,2), (1,4), (4,4).

Practice Exercises: Pure security levels/policies

Exercise 3.2. For the matrix game in **Example 3.1**,

$$A = \underbrace{\left.\begin{bmatrix} 1 & 3 & 3 & -1 \\ 0 & -1 & 2 & 1 \\ -2 & 2 & 0 & 1 \end{bmatrix}\right\} P_1 \text{ choices}}_{P_2 \text{ choices}}$$

show that the pair of policies

$$\pi_2 \equiv P_2 \text{ selects } \begin{cases} \text{column 2 (or 3)} & \text{if } P_1 \text{ selected row 1, leading to a reward of 3} \\ \text{column 3} & \text{if } P_1 \text{ selected row 2, leading to a reward of 2} \\ \text{column 2} & \text{if } P_1 \text{ selected row 3, leading to a reward of 2} \end{cases}$$

and

$$\pi_1 \equiv P_1 \text{ selects row 2 (or 3), leading to a cost of 2}$$

form a Nash equilibrium in the sense of the **Definition 1.1**.

End of Lecture

03 - Zero-Sum Matrix Games

Questions?